Breaking Row Boundaries in a Fully Justified Tile Layout

Devon Rifkin
Yahoo Magazines

October 22, 2013
Revised January 20, 2014

Introduction

When we built the first two Yahoo Magazines, Yahoo Food and Yahoo Tech, we wanted to have a visually striking and immersive tile grid. We found that Flickr's fully-jsutified tile grid had many properties we desired, including leaving images uncropped and a high content density. Our designers chose to base their design on the Flickr grid, but wanted to extend it with additional layout variations.

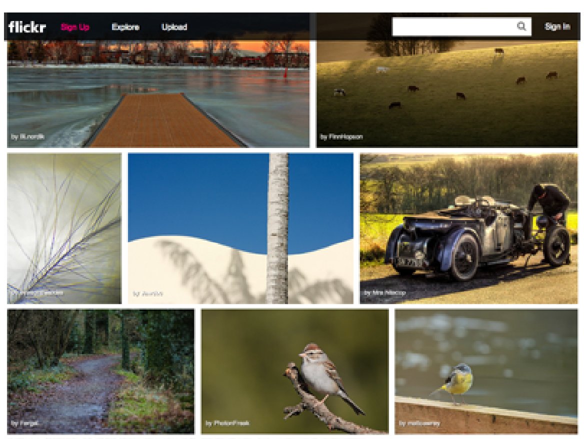

Figure 1: The Flickr grid

They proposed a new row variant where one tile would span the height of two rows, with the rest of the tiles adjusting around it to avoid any gaps. Here is an example of this new row type:

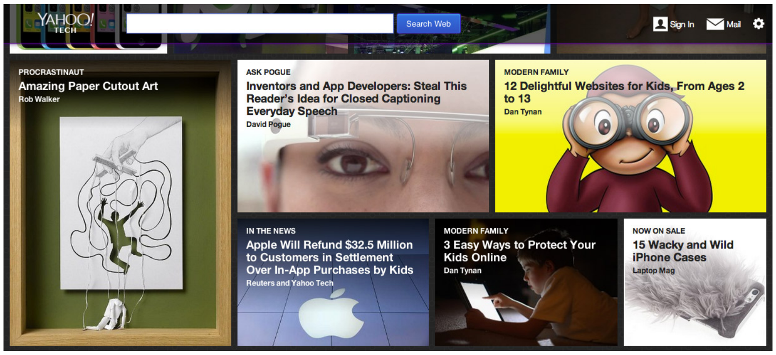

Figure 2: Our row variant, as seen in Yahoo Tech

In order to accomodate this new row type, we developed a simple yet efficient method based on solving a system of equations

Approach

We assume that we have already decided which tiles go in which row. Choosing heuristics for tile placement is a complicated matter and could be the topic of another paper. Instead, we describe how we determine and position the sizes of the tiles after they have been grouped into rows.

Definitions

Fig. 3 shows a labeled diagram of the components of our new row variant.

Based on this diagram, we define the following constants:

$$\begin{aligned}
P &= \text{padding} \\
A_i &= \text{the aspect ratio of the } i\text{th element of row A} \\
B_i &= \text{the aspect ratio of the } i\text{th element of row B} \\
C &= \text{the aspect ratio of the item on the side} \\
N_a &= \text{number of elements in row a} \\
N_b &= \text{number of elements in row b} \\
\\
W &= \text{width of the entire layout}
\end{aligned}$$

Quantities that are known at the time of the layout are capitalized, and aspect ratios are defined as the natural width of an image divided by its natural height.

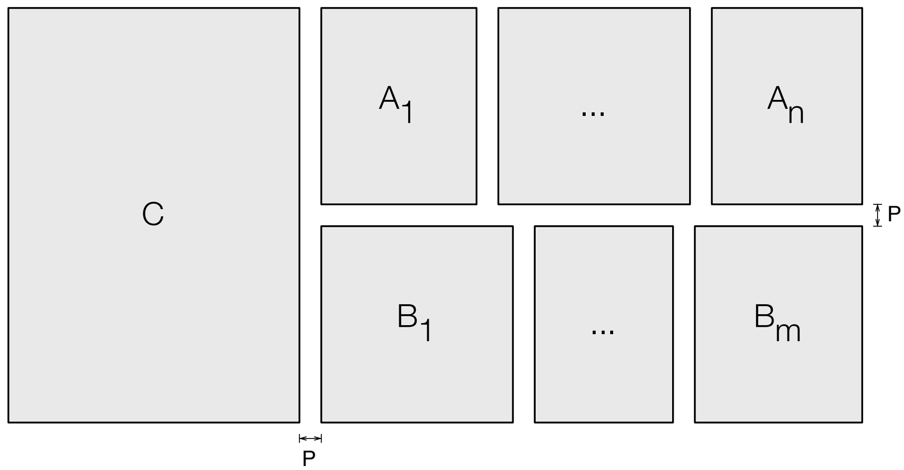

Figure 3: Components of our new row variant

The quantities that our method will solve for are the following:

$$
\begin{aligned}
h_a &= \text{height of row a} \\
h_b &= \text{height of row b} \\
h_c &= \text{height of tile c} \\
\\
w_a &= \text{width of row a} \\
w_b &= \text{width of row b} \\
w_c &= \text{width of tile c} \\
w_{ab} &= w_a = w_b
\end{aligned}
$$

Constraint Equations

Now we will define the constraints of this layout with equations that describe the relationships between the tiles' locations, aspect ratios and sizes.

Width Equations

$$
\begin{aligned}
w_a &= (N_a - 1)P + h_a \sum_{i=0}^{N_a} A_i \\
w_b &= (N_b - 1)P + h_b \sum_{i=0}^{N_b} B_i \\
W &= w_c + P + w_{ab}
\end{aligned}
$$

When we are solving these constraints, we have already decided which tiles are in which row. Therefore, expressions like $(N_a - 1)P$ are constants.

To simplify, we rewrite the following expressions consisting only of constants:

$$\begin{aligned} Q_a &= (N_a - 1)P \\ Q_b &= (N_b - 1)P \\ R_a &= \sum_{i=0}^{N_a} A_i \\ R_b &= \sum_{i=0}^{N_b} B_i \end{aligned}$$

Also, recall that $w_a = w_b$, so we will refer to either as w_{ab}

Simplified Width Equations

$$\begin{aligned} w_{ab} &= Q_a + h_a R_a \\ w_{ab} &= Q_b + h_b R_b \\ W &= w_c + P + w_{ab} \end{aligned}$$

Height Equations

$$\begin{aligned} h_c &= h_a + P + h_b \\ w_c &= C h_c \end{aligned}$$

All Constraint Equations

$$\begin{aligned} w_{ab} - R_a h_a &= Q_a \\ w_{ab} - R_b h_b &= Q_b \\ w_{ab} + w_c &= W - P \\ h_a + h_b - h_c &= -P \\ w_c - C h_c &= 0 \end{aligned}$$

Solving the System of Equations

Now we take these equations and show them in matrix form, so we can solve the system of equations.

$$\begin{bmatrix} 1 & 0 & -R_a & 0 & 0 \\ 1 & 0 & 0 & -R_b & 0 \\ 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & -1 \\ 0 & 1 & 0 & 0 & -C \end{bmatrix} \begin{bmatrix} w_{ab} \\ w_c \\ h_a \\ h_b \\ h_c \end{bmatrix} = \begin{bmatrix} Q_a \\ Q_b \\ W - P \\ -P \\ 0 \end{bmatrix}$$

We write this as an augmented matrix:

$$\left[\begin{array}{ccccc|c} 1 & 0 & -R_a & 0 & 0 & Q_a \\ 1 & 0 & 0 & -R_b & 0 & Q_b \\ 1 & 1 & 0 & 0 & 0 & W-P \\ 0 & 0 & 1 & 1 & -1 & -P \\ 0 & 1 & 0 & 0 & -C & 0 \end{array}\right]$$

We perform a row reduction on this augmented matrix and end up with:

$$\left[\begin{array}{ccccc|c} 1 & 0 & -R_a & 0 & 0 & Q_a \\ 0 & 1 & 0 & 0 & -C & 0 \\ 0 & 0 & 1 & 1 & -1 & -P \\ 0 & 0 & 0 & 1 & -1-\frac{C}{R_a} & \frac{P+Q_a-W}{R_a} - P \\ 0 & 0 & 0 & 0 & -R_b - \frac{R_b}{R_a}C - C & Q_b - Q_a + R_aP + (R_b + R_a)\left(\frac{P+Q_a-W}{R_a} - P\right) \end{array}\right]$$

Since this is a matrix is in row-echelon form, we have a solution to our system of equations. We can represent each of our desired outputs in terms of constants.

Final Equations

The final form of our augmented matrix can be rewritten in equation form:

$$\begin{aligned} h_c &= \frac{Q_b - Q_a + R_aP + (R_b + R_a)\left(\frac{P+Q_a-W}{R_a} - P\right)}{-R_b - \frac{R_b}{R_a}C - C} \\ h_b &= \frac{P + Q_a - W}{R_a} - P + \left(1 + \frac{C}{R_a}\right) h_c \\ h_a &= -P + h_c - h_b \\ w_c &= Ch_c \\ w_{ab} &= Q_a + R_a h_a \end{aligned}$$

These equations give us enough information to lay out our row by positioning and resizing tiles as in the Flickr layout.

Results

This method allows us to lay out rows in the way our designers originally envisioned, and can be seen in action on Yahoo Tech and Yahoo Food. In addition, this method is extremely performant. We compared our closed-form method to a constraint-based layout system[1] and found our method can lay out over 4,000 times more rows per second than the constraint-based system.

[1] We compared our method with a constraint-based layout using a Javascript impementation of Cassowary (https://github.com/slightlyoff/cassowary.js)